Lesson 05

持久化
Persistence

计算机结构

CPU - Memory - Disk

Disk：

持久化存储数据

关机之后 依然在

Memory：

存储数据

程序运行时，任何一个变量都会开在内存里

关机之后，就没了

CPU：

用来计算

1 + 2 -> 3

持久化

关了还能用

文件

网络 发到服务器 服务器存储到文件里

文件操作

09_file

文件基本操作

File

文件读写

Apache Common IO

文件指针

有高级 API 可以直接修改文件内的某个字节区间内的值

序列化
Serialization

问题

对象在内存里 是网状结构

文件 是1维的二进制

网络 是1维的二进制流

序列化 / 反序列化
Serialization / Deserialization

对象 -> 二进制

二进制 -> 对象

拆毛衣 / 织毛衣

分析：

优点：序列化之后的文件大小比较小

缺点：可读性非常低

字符串作为中介的序列化

想法：

对象 -> 字符串 -> 二进制

分析：

字符串 -> 二进制：

有现成的解决方案

会有字符集的问题

对象 -> 字符串：

可以写函数去拼接字符串

案例

10_serial

对象结构：

```
List<Message>

Message
+   text: String
+   id: long
```

字符串转移：

```
message1.id | message1.text
message2.id | message2.text
message3.id | message3.text
```

同步 与 数据更新

同步

同步需求

用户意识 与 界面 同步：

用户点击 添加，界面增加 一个 label

界面 与 文件 同步：

界面增加一个 label 文件 增加一条数据

问题

1. 信息完整：

界面上并没有存储所有的数据

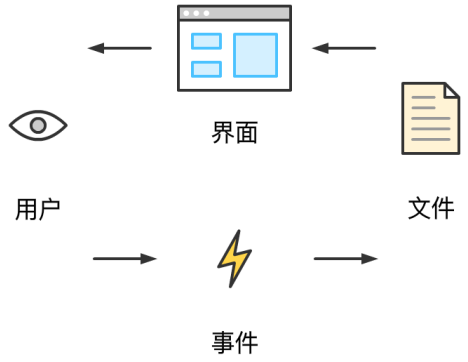
比如 id

2. 同步时机：

是立刻同步，还是保存同步?

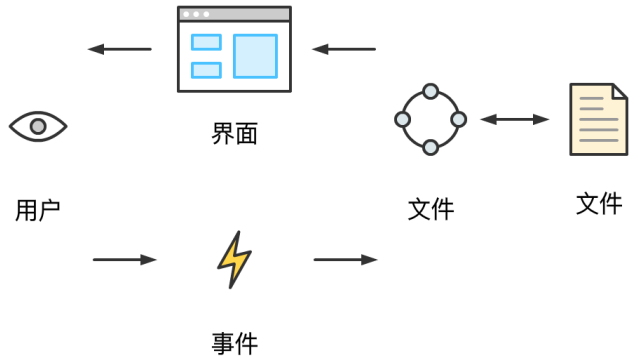
方案

解决方案 1:



能解决问题 1

解决方案 2:



能解决问题 1 2

数据更新